

In the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

1           1.     (Original) A method for maintaining coherency of software  
2 breakpoints in shared memory when debugging a multiple processor  
3 system, the method comprising the steps of:  
4           activating a first debug session associated with a first  
5 processor of a plurality of processors and at least a second debug  
6 session associated with a second processor of the plurality of  
7 processors;  
8           setting a first software breakpoint in a shared memory  
9 location in the first debug session such that all debug sessions  
10 are notified of the setting of the breakpoint; and  
11           clearing the first software breakpoint in the shared memory  
12 location in the second debug session such that all debug sessions  
13 are notified of the clearing of the breakpoint.

1           2.     (Original) The method of Claim 1 wherein the method  
2 comprises the step of creating a software memory map of the memory  
3 usage of a plurality of processors in the system to be debugged.

1           3.     (Original) The method of Claim 2 wherein the step of  
2 setting comprises:  
3           searching the software memory map to find a first plurality of  
4 processors having read access to the shared memory location;  
5           updating a software representation maintained for software  
6 breakpoints for each of the first plurality of processors; and  
7           writing the software breakpoint instruction in the shared  
8 memory location.

1           4.   (Original) The method of Claim 3 wherein the step of  
2 writing comprises a method for selecting a processor to execute the  
3 write, the method comprising the steps of:  
4           if the first processor associated with the first debug session  
5 requesting the setting of the software breakpoint has write access  
6 to the shared memory location  
7           then  
8                   selecting the first processor to perform the write  
9 request;  
10           else performing the following steps a-b:  
11                   a.   searching the software memory map for a second  
12 processor with write access to the shared memory location;  
13                   b.   selecting the second processor to perform the  
14 write request; and  
15           passing the software breakpoint instruction to the selected  
16 processor to be written into the shared memory location.

1           5.   (Original) The method of Claim 4 wherein the step of  
2 passing the software breakpoint instruction comprises the steps of:  
3           searching the software memory map for a second plurality of  
4 processors that have read access to the shared memory location;  
5           broadcasting the write request to the second plurality of  
6 processors; and  
7           performing cache coherency updates in response to the write  
8 request in each of the second plurality of processors as required.

1           6.   (Original) The method of Claim 5 wherein the step of  
2 broadcasting the write request comprises indicating that the write  
3 request is intended for maintaining cache coherency as opposed to a  
4 normal write request.

1       7.   (Original) The method of Claim 6 wherein the step of  
2 performing comprises using cache coherency capabilities, if any, of  
3 a processor in response to the write request intended for  
4 maintaining cache coherency.

1       8.   (Original) The method of Claim 2 wherein the step of  
2 clearing comprises:

3       writing the original instruction stored in a software  
4 representation maintained for software breakpoints into the shared  
5 memory location;

6       searching the software memory map to find a fourth plurality  
7 of processors having read access to the shared memory location; and

8       updating a software representation maintained for software  
9 breakpoints for each of the fourth plurality of processors to  
10 remove the software breakpoint for the shared memory location.

1       9.   (Original) The method of Claim 8 wherein the step of  
2 writing comprises a method for selecting a processor to execute the  
3 write, the method comprising the steps of:

4       if the second processor associated with the second debug  
5 session requesting the clearing of the software breakpoint has  
6 write access to the shared memory location

7           then

8               selecting the second processor to perform the write  
9 request;

10       else performing the following steps a-b:

11       a.   searching the software representation of the memory map  
12 for a third processor with write access to the shared memory  
13 location;

14       b.   selecting the third processor to perform the write  
15 request; and

16        passing the original instruction to the selected processor to  
17 be written into the shared memory location.

1        10. (Original) The method of Claim 9 wherein the step of  
2 passing the original instruction comprises the steps of:  
3        searching the software memory map for a fifth plurality of  
4 processors that have read access to the shared memory location;  
5        broadcasting the write request to the fifth plurality of  
6 processors; and  
7        performing cache coherency updates in response to the write  
8 request in each of the fifth plurality of processors.

1        11. (Original) The method of Claim 10 wherein the step of  
2 broadcasting the write request comprises indicating that the write  
3 request is intended for maintaining cache coherency as opposed to a  
4 normal write request.

1        12. (Original) The method of Claim 11 wherein the step of  
2 performing comprises using cache coherency capabilities, if any, of  
3 a processor in response to the write request intended for  
4 maintaining cache coherency.

1        13. (Original) The method of Claim 1 comprising the step of  
2 stepping over a software breakpoint or running after hitting a  
3 breakpoint in a shared memory location wherein the method for  
4 stepping over the software breakpoint comprises the steps of:  
5        requesting that a software breakpoint in a shared memory  
6 location be stepped over or program execution resumed after hitting  
7 the breakpoint in a third debug session;  
8        "clearing" the software breakpoint in the shared memory location  
9 in the third debug session such that all debug sessions are  
10 notified of the clearing of the breakpoint;

11 stepping a processor associated with the third debug session  
 12 to the instruction after the shared memory location from which the  
 13 software breakpoint was cleared; and  
 14 setting the first software breakpoint in the shared memory  
 15 location in the third debug session such that all debug sessions  
 16 are notified of the setting of the breakpoint.

1 14. (Currently Amended) The method of Claim 13 additionally  
 2 comprising the steps of:

3 searching ~~the~~ a software memory map for a sixth plurality of  
 4 processors having read access to the shared memory location; and  
 5 halting all processors in the sixth plurality of processors  
 6 after the step of requesting and prior to the step of clearing.

1 15. (Currently Amended) The method of Claim 13 wherein the  
 2 step of setting comprises:

3 searching ~~the~~ a software memory map to find a seventh  
 4 plurality of processors having read access to the shared memory  
 5 location;

6 updating a software representation maintained for software  
 7 breakpoints for each of the seventh plurality of processors; and

8 writing the software breakpoint instruction in the shared  
 9 memory location.

1 16. (Currently Amended) The method of Claim 13 wherein the  
 2 step of clearing comprises:

3 writing the original instruction stored in a software  
 4 representation maintained for software breakpoints in the shared  
 5 memory location;

6 ~~searching the~~ a software memory map to find an eighth  
 7 plurality of processors having read access to the shared memory  
 8 location; and

9 updating a software representation maintained for software  
10 breakpoints for each of the eighth plurality of processors to  
11 remove the software breakpoint for the shared memory location.

1 17. (Original) A software development system, comprising:  
2 a memory storage system holding a software development tool  
3 program;  
4 a host computer connected to the memory storage system, the  
5 host computer operable to execute the software development tool  
6 program;  
7 a test port for connecting to a target hardware system, the  
8 hardware system being comprised of multiple processors with common  
9 shared memory and operable to execute an application program; and  
10 wherein the software development tool is operable to support  
11 debugging of the application program executing on the target  
12 hardware system using a method for maintaining coherency of  
13 software breakpoints in shared memory when debugging a multiple  
14 processor system, the method comprising the steps of:  
15 activating a first debug session associated with a first  
16 processor of a plurality of processors and at least a second debug  
17 session associated with a second processor of the plurality of  
18 processors;  
19 setting a first software breakpoint in a shared memory  
20 location in the first debug session such that all debug sessions  
21 are notified of the setting of the breakpoint; and  
22 clearing the first software breakpoint in the shared  
23 memory location in the second debug session such that all debug  
24 sessions are notified of the clearing of the breakpoint.

18. (Original) The software development system of Claim 17  
wherein the method comprises the step of creating a software memory

3 map of the memory usage of a plurality of processors in the system  
4 to be debugged.

1 19. (Original) The software development system of Claim 18  
2 wherein the step of setting comprises:

3 searching the software memory map to find a first plurality of  
4 processors having read access to the shared memory location;

5 updating a software representation maintained for software  
6 breakpoints for each of the first plurality of processors; and

7 writing the software breakpoint instruction in the shared  
8 memory location.

1 20. (Original) The software development system of Claim 19  
2 wherein the step of writing comprises a method for selecting a  
3 processor to execute the write, the method comprising the steps of:

4 if the first processor associated with the first debug session  
5 requesting the setting of the software breakpoint has write access  
6 to the shared memory location

7 then

8 selecting the first processor to perform the write  
9 request;

10 else performing the following steps a-b:

11 a. searching the software memory map for a second  
12 processor with write access to the shared memory location;

13 b. selecting the second processor to perform the  
14 write request; and

15 passing the software breakpoint instruction to the selected  
16 processor to be written into the shared memory location.

21. (Original) A digital system, comprising:

22 multiple processors with common shared memory for executing an  
3 application program; and

wherein the application program was developed with a software development system using a method for maintaining coherency of software breakpoints in shared memory when debugging a multiple processor system, the method comprising the steps of:

activating a first debug session associated with a first processor of a plurality of processors and at least a second debug session associated with a second processor of the plurality of processors;

setting a first software breakpoint in a shared memory location in the first debug session such that all debug sessions are notified of the setting of the breakpoint; and

clearing the first software breakpoint in the shared memory location in the second debug session such that all debug sessions are notified of the clearing of the breakpoint.

22. (Original) The digital system of Claim 21 wherein the method comprises the step of creating a software memory map of the memory usage of a plurality of processors in the system to be debugged.

23. (Original) The digital system of Claim 22 wherein the step of setting comprises:

searching the software memory map to find a first plurality of processors having read access to the shared memory location;

updating a software representation maintained for software breakpoints for each of the first plurality of processors; and

writing the software breakpoint instruction in the shared memory location.

24. (Original) The digital system of Claim 23 wherein the step of writing comprises a method for selecting a processor to

execute the write, the method comprising the steps of:



4       if the first processor associated with the first debug session  
5     requesting the setting of the software breakpoint has write access  
6     to the shared memory location  
7       then  
8             selecting the first processor to perform the write  
9     request;  
10       else performing the following steps a-b:  
11           a.   searching the software representation of the  
12     memory map for a second processor with write access to the shared  
13     memory location;  
14           b.   selecting the second processor to perform the  
15     write request; and  
16       passing the software breakpoint instruction to the selected  
17     processor to be written into the shared memory location.